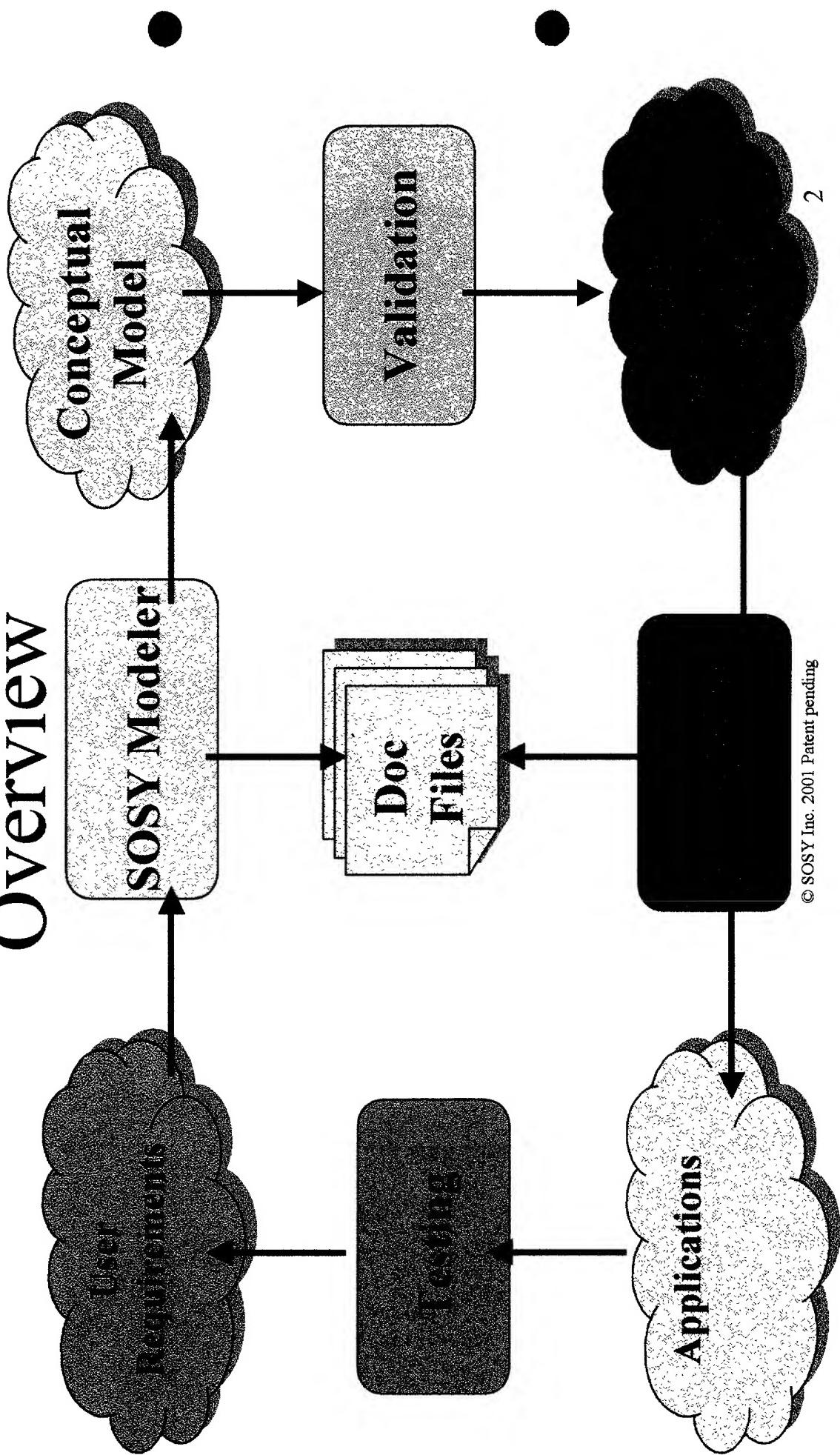


Summary

Overview



CARE Technologies, S.A.

Conceptual Modeling Phase

Index

- Intro
- Overview
- Phase 0. Requirements elicitation.
- Phase 1. Classes identification.
- Phase 2. Relationships between classes.
- Phase 3. Filling classes' details.

Index

- Phase 4. Express evaluations.
- Phase 5. Agent relationships.
- Phase 6. State Transition Diagram.
- Phase 7. Presentation Model.

Intro

- Conceptual Modeling Phase is a process of systematically & precisely description of the system to build, using:
 - Graphical UML compliant diagrams.
 - Constraints and semantics in a formal non-ambiguous language.
 - This phase is assisted by an integrated Modeler tool.

Overview



Requirements

- Specifications
- Documents
- Interviews
- Reports
- Other info. sources

Conceptual Model

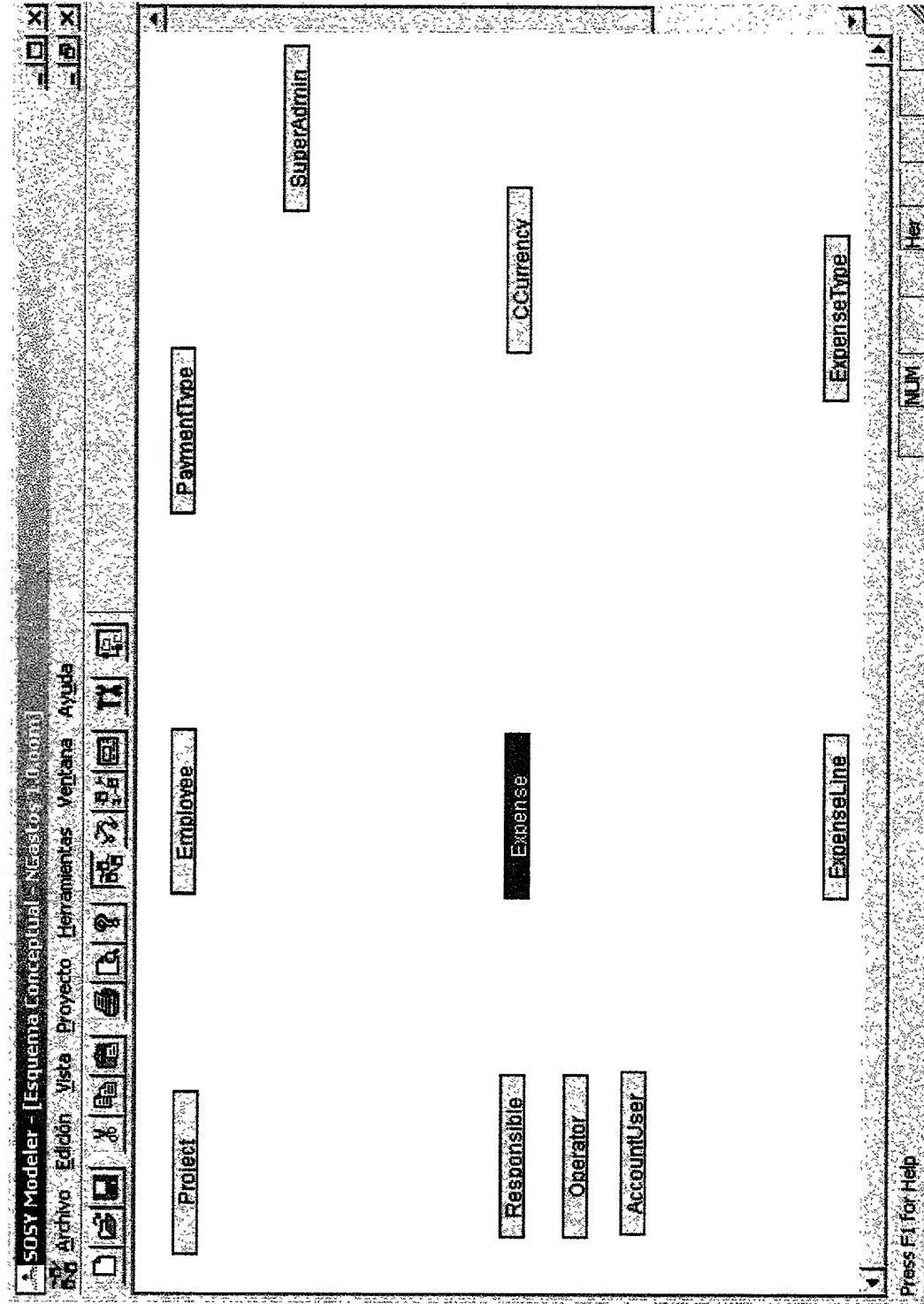
- Classes
- Relationships
- Attributes
- Services
- ...

Expressed in a non-ambiguous language.

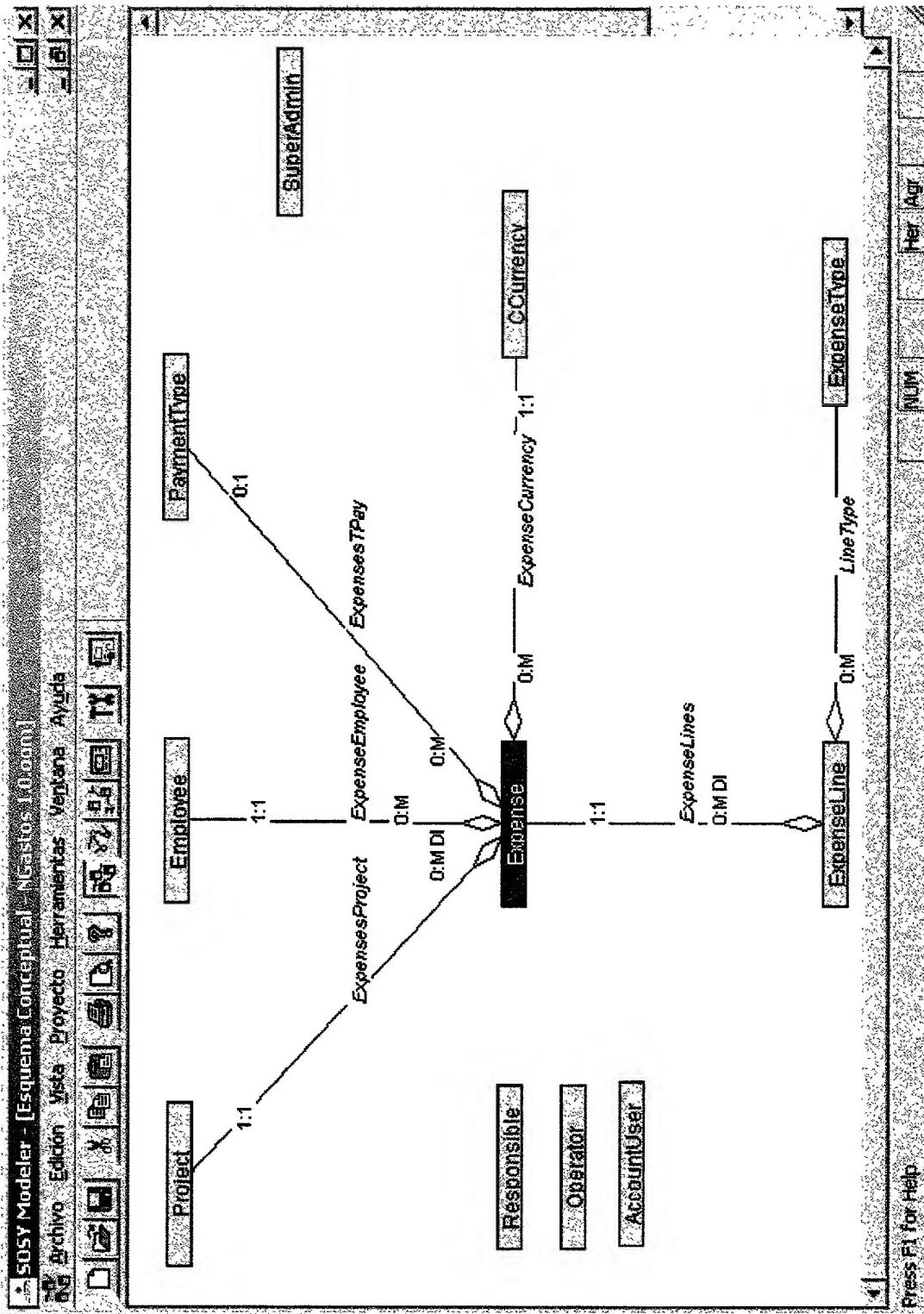
Phase 0. Requirement elicitation.

- Gathering the system requirements.
 - By meetings & interviews with customers, experts and final users.
 - By collecting reports, or documents expressing the system how-to and using tools.
 - Obtaining a coherent set of information as input to the next phase.

Phase 1. Classes identification.



Phase 2. Relationships between classes.



Phase 3. Filling classes' details.

X Clase

Attributos | Servicios | Denominaciones | Plastificaciones | Agentes | Transacciones | Relaciones | Generatividades |

Atributos

Nombre	Tipo Atributo	Constante	Tipo dato	Id	Tamaño	Valor default	Padrón al crear	Nulos	Único
PresentDate	Variable		Date			today()	Si	No	
Status	Variable		Int			0	No	No	
Cause	Variable		String				Sí	No	
AuthoDate	Variable		Date			NULL	No	Sí	
AuthoComments	Variable		String			NULL	No	Sí	
PaymentDate	Variable		Date			NULL	No	Sí	
PayComments	Variable		Date			NULL	No	Sí	
TotalExpenses	Derivado		Real				No	No	
TotalExpensesCur	Derivado		Real				No	No	
Advances	Variable		Real				No	No	
AdvancesCur	Derivado		Real				No	No	
Exchange	Variable		Real				No	No	
Balance	Derivado		Real				No	No	
BalanceCur	Derivado		Real				No	No	

Nombre: Alias:

Observaciones:

Tipo Atributo: Tipo Dato:

Notes >

Información Física

Close

Phase 3. Filling classes' details.

Clase	
atributos servicios	Definiciones Restricciones Agencias Transacciones Relaciones Generalidades
Atributo	<input type="text" value="Balance"/>
Formulas de Derivación	<input type="text" value="Condición"/>
Condición	<input type="text" value="Fórmula"/>
Fórmula	<input type="text" value="Total Expenses - Advances"/>
Observaciones	<input type="text" value=""/>
Clase:	<input type="text" value="Expense"/>
Acción:	<input type="text" value="Aceptar"/>
Comisión:	<input type="text" value=""/>

Phase 3. Filling classes' details.

Phase 3. Filling classes' details.

Clase		Atributos Servicios Devivencias Restricciones Agentes Transacciones Relaciones Generalidades																													
<input style="width: 100px; height: 25px; margin-bottom: 5px;" type="button" value="Transaction"/> <input style="width: 100px; height: 25px;" type="button" value="DELETEALL"/> <input style="width: 100px; height: 25px; margin-bottom: 5px;" type="button" value="Formulas"/>		<p>FOR ALL Lines DQ Lines deleteLine[Lines]. delExpense[THIS]</p>																													
		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Acción</td> <td colspan="3" style="text-align: right;">C Gr and C Acción</td> </tr> <tr> <td>Expense</td> <td colspan="3"> <input type="checkbox"/> Details </td> </tr> <tr> <td>Agents</td> <td style="width: 15%;">Servicio:</td> <td colspan="2"> <input type="checkbox"/> approve </td> </tr> <tr> <td></td> <td>Payámetros:</td> <td colspan="2"> <input type="checkbox"/> P. thisExpense </td> </tr> <tr> <td></td> <td>Iniciar:</td> <td colspan="2"> <input type="checkbox"/> Crea Patrón en la Transacción </td> </tr> <tr> <td colspan="4">Observaciones</td> </tr> <tr> <td colspan="4"> <input style="width: 100px; height: 25px; margin-top: 10px;" type="button" value="Expense"/> </td> </tr> </table>		Acción	C Gr and C Acción			Expense	<input type="checkbox"/> Details			Agents	Servicio:	<input type="checkbox"/> approve			Payámetros:	<input type="checkbox"/> P. thisExpense			Iniciar:	<input type="checkbox"/> Crea Patrón en la Transacción		Observaciones				<input style="width: 100px; height: 25px; margin-top: 10px;" type="button" value="Expense"/>			
Acción	C Gr and C Acción																														
Expense	<input type="checkbox"/> Details																														
Agents	Servicio:	<input type="checkbox"/> approve																													
	Payámetros:	<input type="checkbox"/> P. thisExpense																													
	Iniciar:	<input type="checkbox"/> Crea Patrón en la Transacción																													
Observaciones																															
<input style="width: 100px; height: 25px; margin-top: 10px;" type="button" value="Expense"/>																															

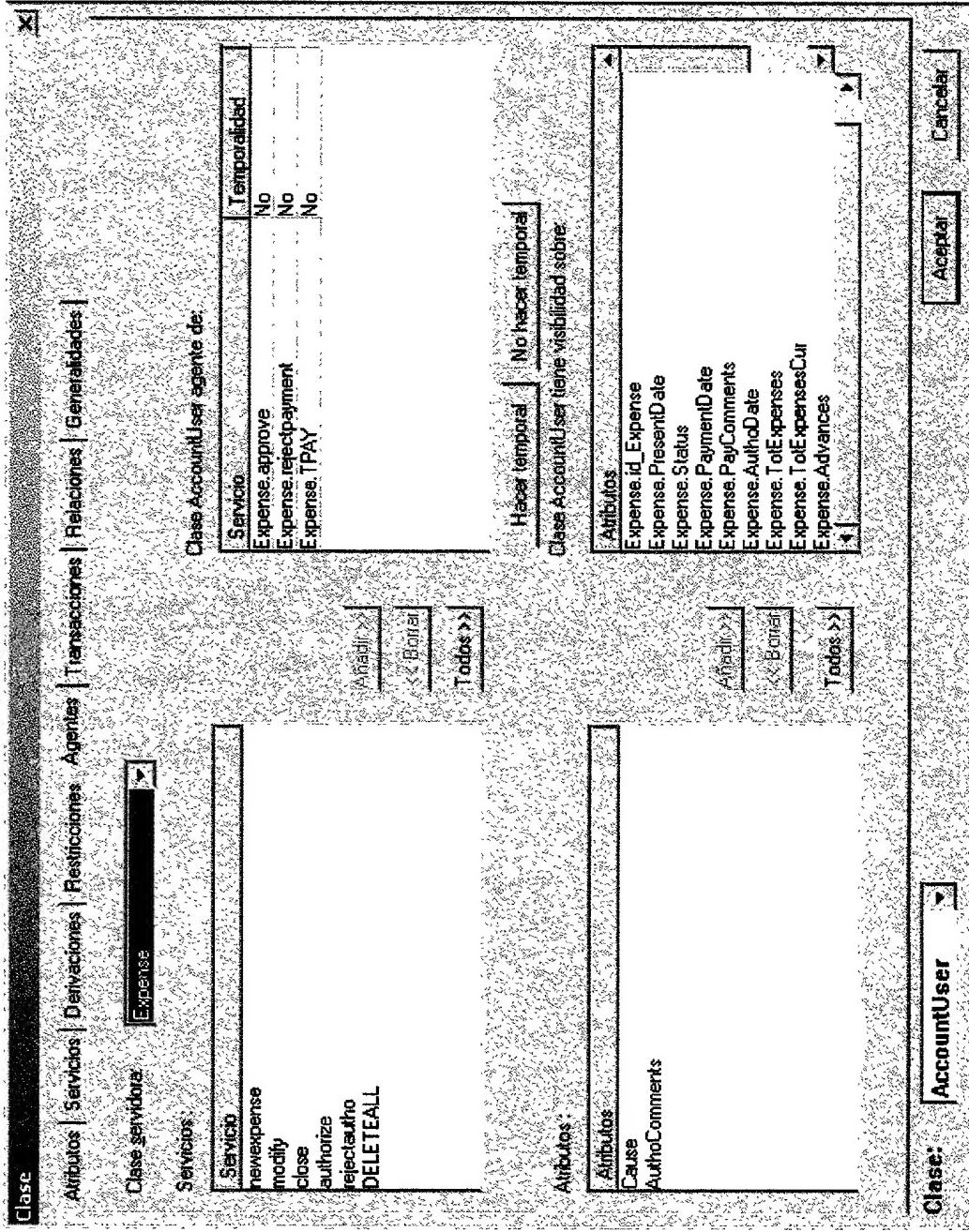
Phase 3. Filling classes' details.

Clase	
Atributos	Servicios Relaciones Restricciones Agencias Transacciones Relaciones Generalidades
Estáticas	<input type="text"/> Exchange > 0 <input type="text"/> Exchange must be greater than zero
Dinámicas	<input type="text"/> Condicion1 <input type="text"/> Condicion2 <input type="text"/> Oper1 temporal <input type="text"/> Oper2 temporal <input type="text"/> Mensaje De Error
Clase	<input type="text"/> Expense <input type="checkbox"/> Aceptar <input type="checkbox"/> Cancelar

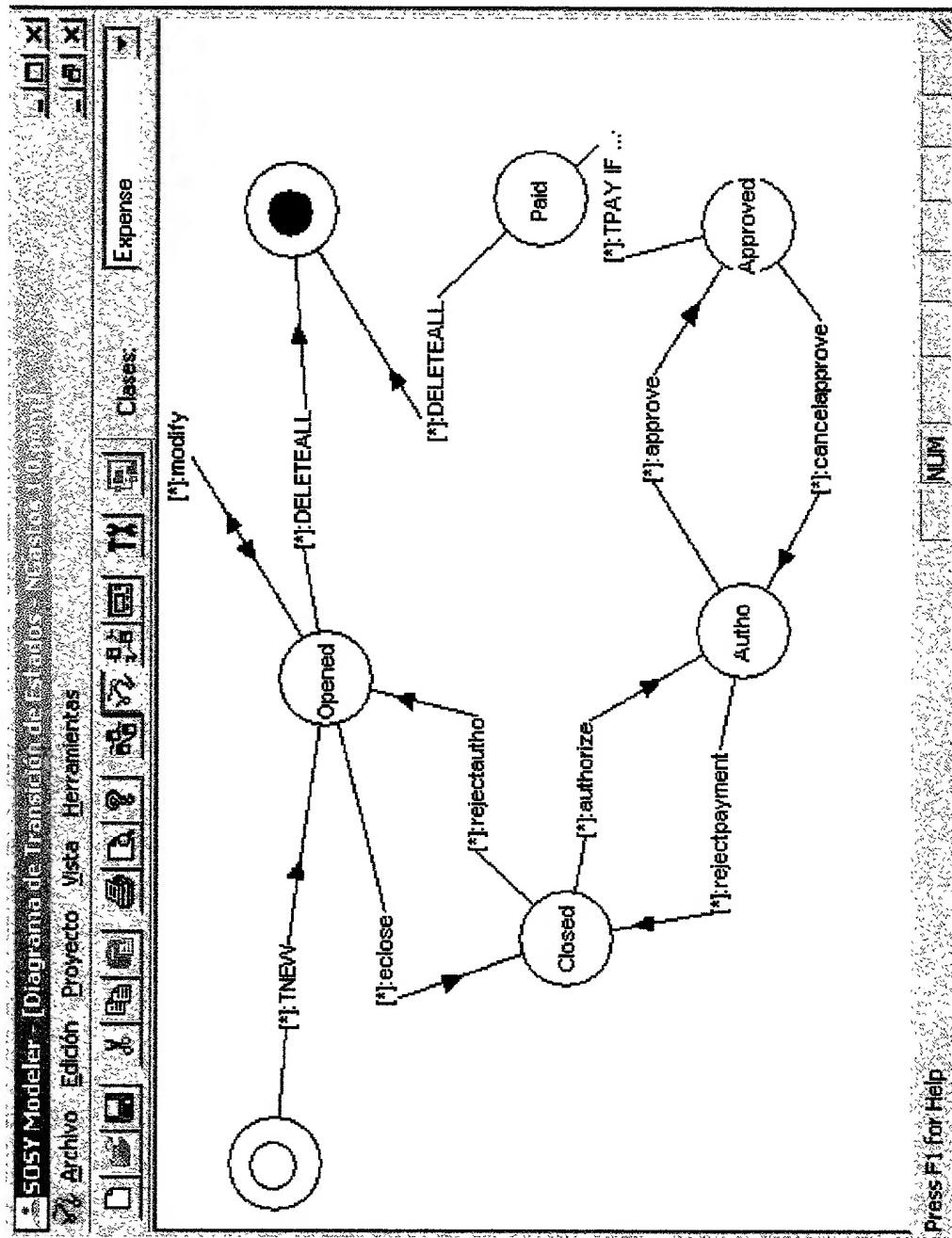
Phase 4. Express evaluations.

Modelo Funcional															
Clase:	<input type="text" value="Expense"/>	Attributo:	<input type="text" value="Cause"/>												
		<input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/>													
<table border="1"> <tr> <td>Evento</td> <td>Efecto</td> <td>Condición</td> <td></td> </tr> <tr> <td>modify</td> <td>=</td> <td>P_Cause</td> <td></td> </tr> </table>				Evento	Efecto	Condición		modify	=	P_Cause					
Evento	Efecto	Condición													
modify	=	P_Cause													
<table border="1"> <tr> <td>Categoría</td> <td><input checked="" type="radio"/> De Estado</td> <td><input type="radio"/> De Situación</td> </tr> <tr> <td>Cardinal</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> </table>				Categoría	<input checked="" type="radio"/> De Estado	<input type="radio"/> De Situación	Cardinal	<input type="radio"/>	<input type="radio"/>						
Categoría	<input checked="" type="radio"/> De Estado	<input type="radio"/> De Situación													
Cardinal	<input type="radio"/>	<input type="radio"/>													
<table border="1"> <tr> <td>Detalles de Evaluación</td> <td></td> </tr> <tr> <td>Evento</td> <td><input type="text" value="modify"/></td> </tr> <tr> <td colspan="2"> <input type="checkbox"/> Intent para el resto de atributos </td> </tr> <tr> <td colspan="2"> <input type="checkbox"/> Condición de evaluación </td> </tr> <tr> <td colspan="2"> <input type="checkbox"/> Efecto del evento </td> </tr> <tr> <td colspan="2"> <input type="checkbox"/> P_Cause </td> </tr> </table>				Detalles de Evaluación		Evento	<input type="text" value="modify"/>	<input type="checkbox"/> Intent para el resto de atributos		<input type="checkbox"/> Condición de evaluación		<input type="checkbox"/> Efecto del evento		<input type="checkbox"/> P_Cause	
Detalles de Evaluación															
Evento	<input type="text" value="modify"/>														
<input type="checkbox"/> Intent para el resto de atributos															
<input type="checkbox"/> Condición de evaluación															
<input type="checkbox"/> Efecto del evento															
<input type="checkbox"/> P_Cause															

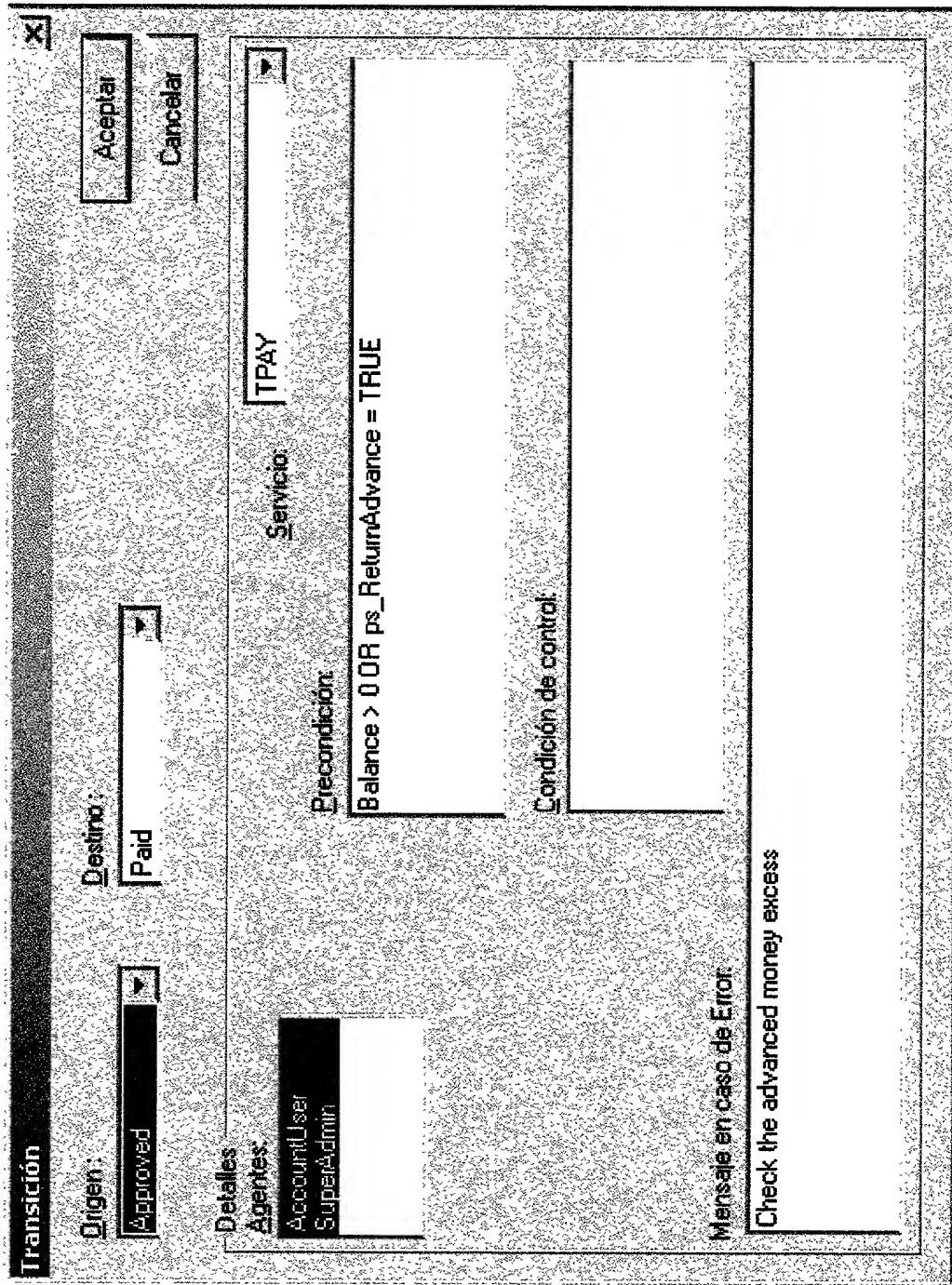
Phase 5. Agent relationships.



Phase 6. State Transition Diagram.



Phase 6. STD Preconditions



Phase 7. Presentation Model.

Conjunto de visualización																																								
Nombre: <input type="text" value="CV_Expense"/>	<input type="button" value="Limpiar"/>																																							
<input type="button" value="Atributos a visualizar"/>																																								
<table border="1"> <thead> <tr> <th>Atributo</th> <th>Tipo dato</th> <th></th> </tr> </thead> <tbody> <tr> <td>Project.ProjectName</td> <td>String</td> <td><input type="button" value="Añadir >"/></td> </tr> <tr> <td>Employee.EmpName</td> <td>String</td> <td><input type="button" value="Eliminar >>"/></td> </tr> <tr> <td>Employee.EmpSur...</td> <td>String</td> <td><input type="button" value="Subir"/></td> </tr> <tr> <td>Status</td> <td>Int</td> <td><input type="button" value="Bajar"/></td> </tr> <tr> <td>AuthoDDate</td> <td>Date</td> <td><input type="button" value="Agregat"/></td> </tr> <tr> <td>PaymentDDate</td> <td>Date</td> <td></td> </tr> <tr> <td>TotalExpenses</td> <td>Real</td> <td></td> </tr> <tr> <td>Balance</td> <td>Real</td> <td></td> </tr> </tbody> </table>		Atributo	Tipo dato		Project.ProjectName	String	<input type="button" value="Añadir >"/>	Employee.EmpName	String	<input type="button" value="Eliminar >>"/>	Employee.EmpSur...	String	<input type="button" value="Subir"/>	Status	Int	<input type="button" value="Bajar"/>	AuthoDDate	Date	<input type="button" value="Agregat"/>	PaymentDDate	Date		TotalExpenses	Real		Balance	Real													
Atributo	Tipo dato																																							
Project.ProjectName	String	<input type="button" value="Añadir >"/>																																						
Employee.EmpName	String	<input type="button" value="Eliminar >>"/>																																						
Employee.EmpSur...	String	<input type="button" value="Subir"/>																																						
Status	Int	<input type="button" value="Bajar"/>																																						
AuthoDDate	Date	<input type="button" value="Agregat"/>																																						
PaymentDDate	Date																																							
TotalExpenses	Real																																							
Balance	Real																																							
<input type="button" value="Atributos"/>																																								
<table border="1"> <thead> <tr> <th>Atributo</th> <th>Tipo dato</th> <th></th> </tr> </thead> <tbody> <tr> <td>Cause</td> <td>String</td> <td><input type="button" value="Añadir >"/></td> </tr> <tr> <td>AuthoDDate</td> <td>Date</td> <td><input type="button" value="Eliminar >>"/></td> </tr> <tr> <td>AuthoComments</td> <td>String</td> <td><input type="button" value="Subir"/></td> </tr> <tr> <td>PaymentDDate</td> <td>Date</td> <td><input type="button" value="Bajar"/></td> </tr> <tr> <td>PayComments</td> <td>String</td> <td><input type="button" value="Agregat"/></td> </tr> <tr> <td>TotalExpenses</td> <td>Real</td> <td></td> </tr> <tr> <td>TotalExpensesCur</td> <td>Real</td> <td></td> </tr> <tr> <td>Advances</td> <td>Real</td> <td></td> </tr> <tr> <td>AdvancesCur</td> <td>Real</td> <td></td> </tr> <tr> <td>Exchange</td> <td>Real</td> <td></td> </tr> <tr> <td>Balance</td> <td>Real</td> <td></td> </tr> <tr> <td>BalanceCur</td> <td>Real</td> <td></td> </tr> </tbody> </table>		Atributo	Tipo dato		Cause	String	<input type="button" value="Añadir >"/>	AuthoDDate	Date	<input type="button" value="Eliminar >>"/>	AuthoComments	String	<input type="button" value="Subir"/>	PaymentDDate	Date	<input type="button" value="Bajar"/>	PayComments	String	<input type="button" value="Agregat"/>	TotalExpenses	Real		TotalExpensesCur	Real		Advances	Real		AdvancesCur	Real		Exchange	Real		Balance	Real		BalanceCur	Real	
Atributo	Tipo dato																																							
Cause	String	<input type="button" value="Añadir >"/>																																						
AuthoDDate	Date	<input type="button" value="Eliminar >>"/>																																						
AuthoComments	String	<input type="button" value="Subir"/>																																						
PaymentDDate	Date	<input type="button" value="Bajar"/>																																						
PayComments	String	<input type="button" value="Agregat"/>																																						
TotalExpenses	Real																																							
TotalExpensesCur	Real																																							
Advances	Real																																							
AdvancesCur	Real																																							
Exchange	Real																																							
Balance	Real																																							
BalanceCur	Real																																							
<input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/>																																								

Clase: Expense

Phase 7. Presentation Model.

Filtro:	<input type="text" value="fl_Expense"/>	Alias:	<input type="text" value="Expense Reports"/>	Limpiar	Borrar																														
Fórmula:	<pre>Project = vf_Project AND Employee = vf_Employee AND PresentDate >= vf_DateInIssue AND PresentDate <= vf_DateEndIssue AND AuthoDate >= vf_DateInApp AND AuthoDate <= vf_DateEndApp AND PaymentDate >= vf_DateInPay AND PaymentDate <= vf_DateEndPay AND</pre>																																		
Observ:	<input type="button" value="Nuevo"/> <input type="button" value="Modificar"/> <input type="button" value="Borrar"/>																																		
Variables:	<table border="1"> <thead> <tr> <th>Nombre</th> <th>Alias</th> <th>Tipo dato</th> <th>Tipo estilo</th> <th>Estilo</th> </tr> </thead> <tbody> <tr> <td>vf_Project</td> <td>Project</td> <td>Project</td> <td>Sel. Población</td> <td>Nueva</td> </tr> <tr> <td>vf_Employee</td> <td>Employee</td> <td>Employee</td> <td>Sel. Población</td> <td>Modificar</td> </tr> <tr> <td>vf_DateInIssue</td> <td>InitialIssuing Date</td> <td>Date</td> <td></td> <td>Borrar</td> </tr> <tr> <td>vf_DateEndIssue</td> <td>FinalIssuing Date</td> <td>Date</td> <td></td> <td></td> </tr> <tr> <td>vf_DateInApp</td> <td>InitialApproving D...</td> <td>Date</td> <td></td> <td></td> </tr> </tbody> </table>					Nombre	Alias	Tipo dato	Tipo estilo	Estilo	vf_Project	Project	Project	Sel. Población	Nueva	vf_Employee	Employee	Employee	Sel. Población	Modificar	vf_DateInIssue	InitialIssuing Date	Date		Borrar	vf_DateEndIssue	FinalIssuing Date	Date			vf_DateInApp	InitialApproving D...	Date		
Nombre	Alias	Tipo dato	Tipo estilo	Estilo																															
vf_Project	Project	Project	Sel. Población	Nueva																															
vf_Employee	Employee	Employee	Sel. Población	Modificar																															
vf_DateInIssue	InitialIssuing Date	Date		Borrar																															
vf_DateEndIssue	FinalIssuing Date	Date																																	
vf_DateInApp	InitialApproving D...	Date																																	
Tipo:	<input type="radio"/> Nombre <input checked="" type="radio"/> Estilo de interfaz <input type="radio"/> Alias <input type="radio"/> Estilo de selección <input type="radio"/> Tipo de dato																																		
Clase:	<input type="radio"/> Expense <input type="radio"/> Acepta <input type="radio"/> Cancelar																																		

Conceptual Model Validation

CARE Technologies, S.A.

Index

- Intro
- Overview
- Validation Degrees
 - Partial Validation
 - Total Validation

Index

- **Validation Types**
 - Elements of the Conceptual Model
 - Formulas of the Conceptual Model (Syntax)
- **Validation Trees**
 - Nodes
 - Leaves
- **Example**

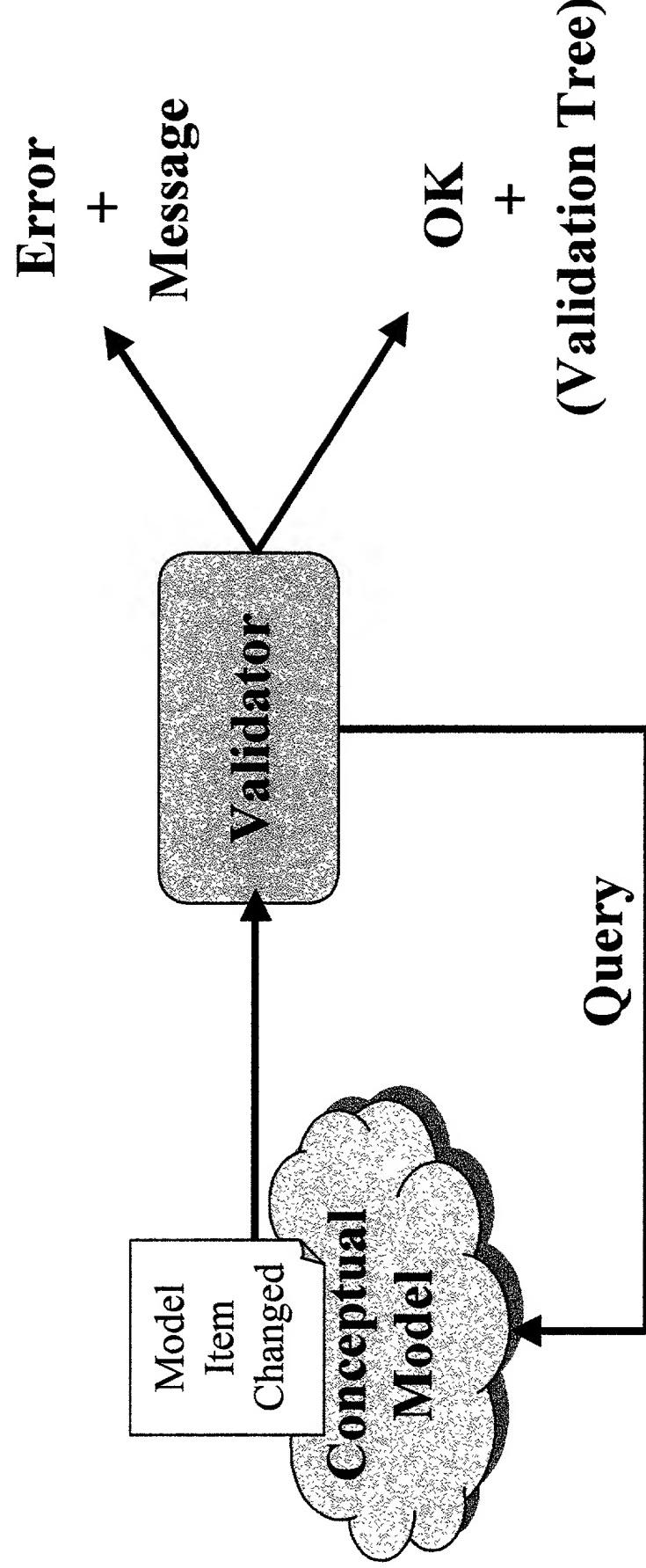
Intro

- Conceptual Model Validation is the process by which a conceptual model or a modification of it is proven to be valid:
 - Correct
 - Non Ambiguous
 - Non Contradictory
 - Complete
 - Every concept is fully specified
- Validation process checks the representation of requirements in Formal Specification Language to be valid

Validation Degrees

- Partial Validation
 - That of a single element of the Conceptual Model.
 - Happens whenever an element is added, modified or deleted.

Partial Validation Overview

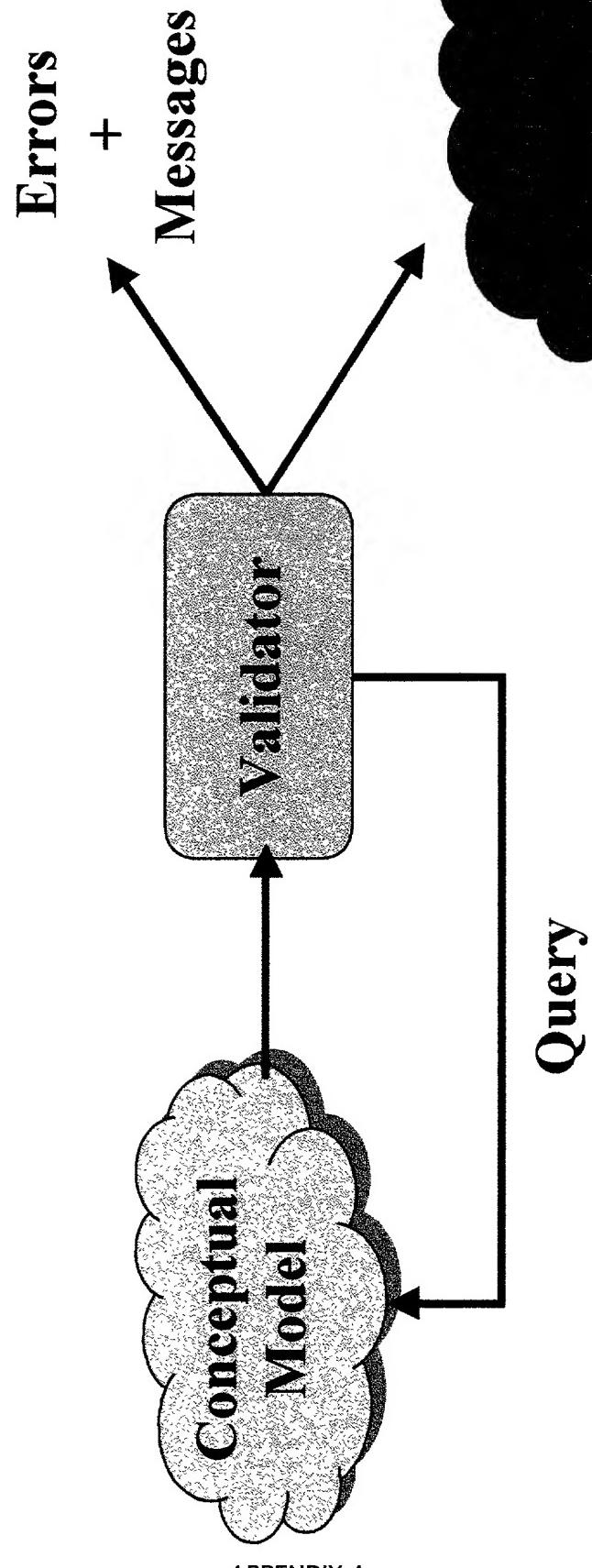


APPENDIX A

Validation Degrees

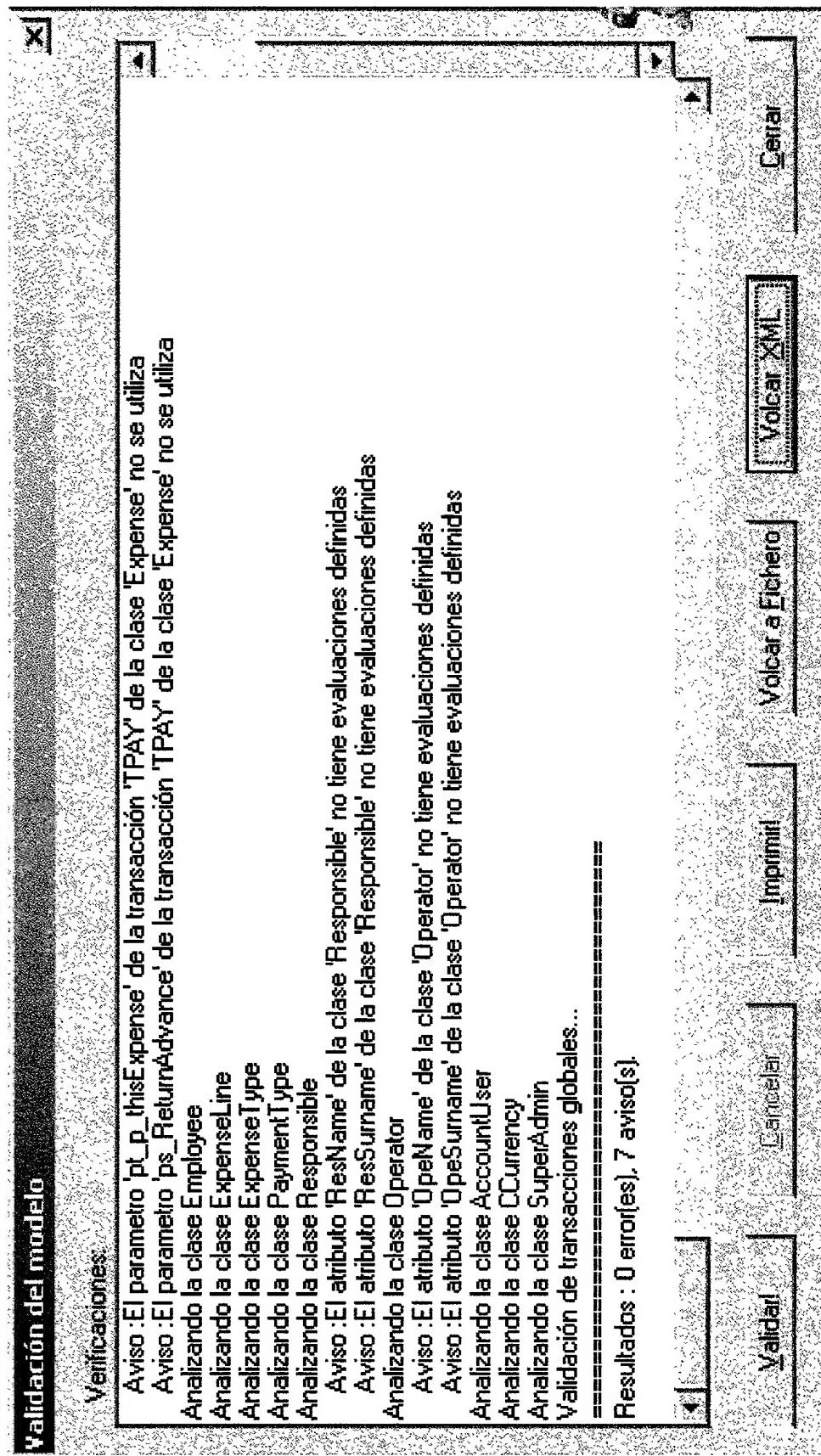
- Total Validation
 - That of the whole Conceptual Model.
 - Happens by request.
 - Must happen prior to any translation process.
 - Takes advantage of partial validations already performed.

Total Validation Overview



APPENDIX A

Total Validation Example



Validation Types

- Elements of the Conceptual Model
 - Ensure the properties of an element (except formulas) are correct and complete.
 - Conditions that must hold depend on the type of element and the property being validated.
 - Examples:
 - Class Name is unique in a Conceptual Model.
 - Attribute Name is unique in its Class (but not in a Conceptual Model)

Validation Types

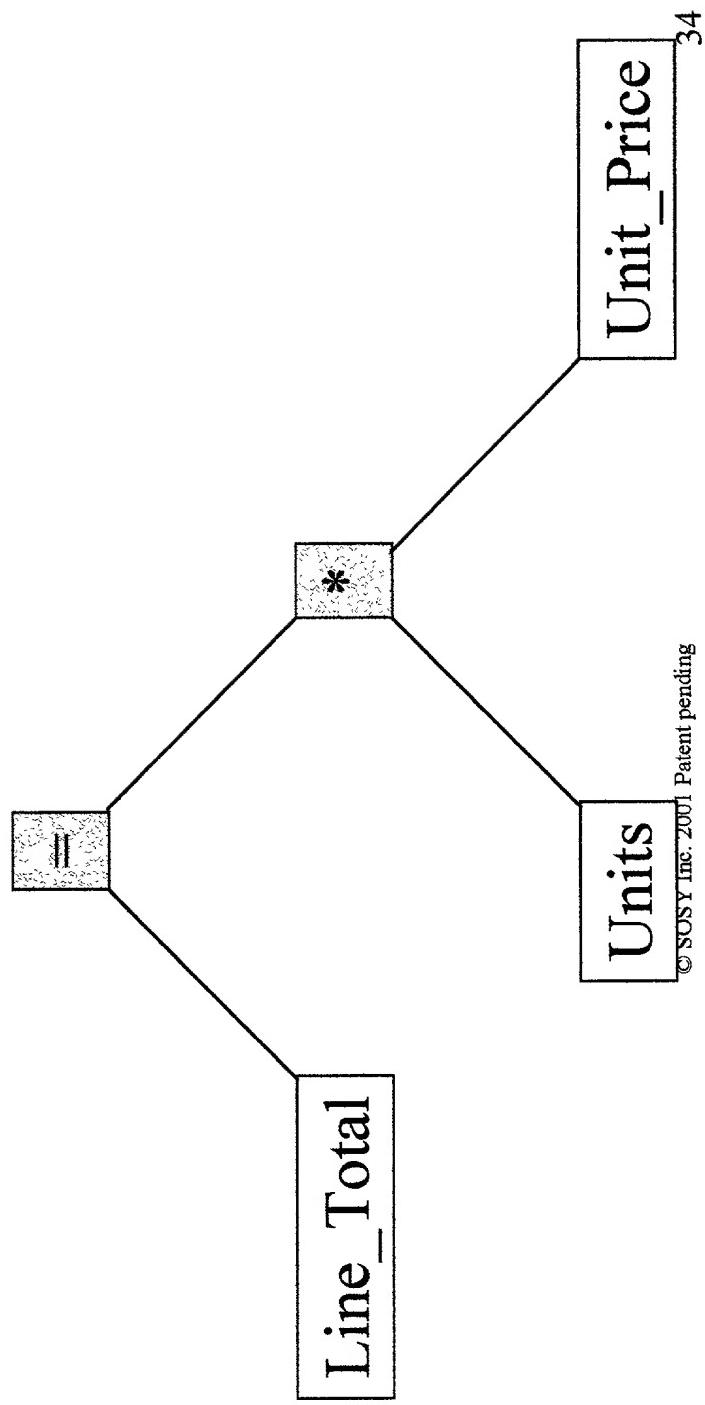
- **Formulas of the Conceptual Model**
 - Ensure the formulas of the Conceptual Model are correct and complete.
 - Syntactical and Semantical Validation according to an extended Formal Specification Language grammar.
 - Input:
 - Formula expression
 - Formula Type (precondition, valuation, ...etc.)
 - Formula Context (class name, service name, ...etc.)
 - Output:
 - Error Message (validation did not pass)
 - Validation Tree (validation passed)

Validation Trees

- Binary Tree representation of a correct formula.
- Tree consists of Nodes and Leaves.
 - Nodes
 - Represent operators
 - Can have one or two “branches” (binary)
 - Branches can again be nodes or leaves
 - Leaves
 - Represent operands
 - Have no branches

Example

- $\text{Line_Total} = \text{Units} * \text{Unit_Price}$



© SOSY Inc. 2001. Patent pending

CARE Technologies, S.A.

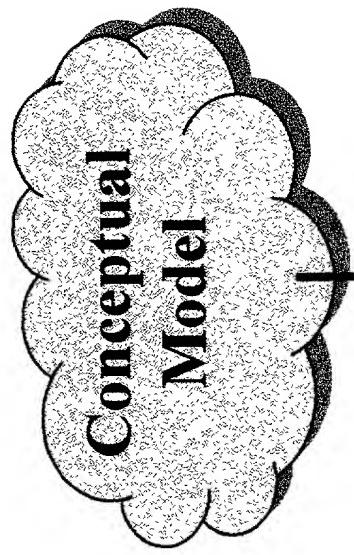
Documentation Translation

Index

- Intro
- Overview
- Output Detail
 - Document Types
 - Document Formats
- Translation
 - CM Subset of Interest
 - Translation Process
 - Remarks
- Example

Intro

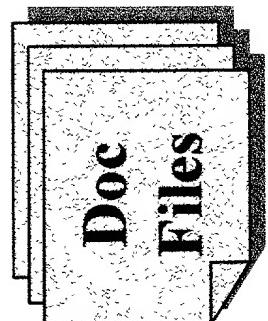
- Documentation Translation is the process to obtain, from a Conceptual Model, documentation on the system it represents.
- Documentation can have several degrees of detail and be focused on different aspects, thus obtaining different documentation formats from the same Conceptual Model.



Overview

Document Type

- Help
 - Full
 - General
 - User Help Manual
 - Project Report
- Test Report
- Multifile HTML
- Single File HTML
- ASCII Text
- LaTeX
- RTF
- Compiled HTML



Document Format

Output Detail

- **Document Types**
 - Help
 - Description of each Class, its Attributes, Services and Population Selection Filters.
 - Full
 - Full description of a Conceptual Model
 - Aimed at analysts.
 - General
 - Description of each Class Attributes, Identification Function, Services, Aggregation Relationships and Specialization Relationships.

Output Detail

- **Document Types**
 - User Help Manual
 - Both Help Manual and Contextual Help (F1 key).
 - Intended for Operation Manual.
 - Integration with User Interface applications.
 - Project Report
 - Description of each Class Attributes and Services.
 - Test Report
 - Description of each Class Services.
 - Intended for Testing purposes.

Output Detail

- Document Formats
 - Multifile HTML
 - One HTML page per concept.
 - Recommended for navigable help.
 - Single File HTML
 - One single HTML page.
 - Recommended for printing.
 - ASCII Text
 - Single, plain ASCII text file.

Output Detail¹

- Document Formats
 - LaTeX
 - Single, LaTeX text file.
 - RTF
 - Single, RTF text file.
 - Compiled HTML
 - Same as Multifile HTML plus header files to be used by HTML Help Workshop compiler.
 - Recommended for contextual help.
 - Searching and Indexing facilities usage from browsers.

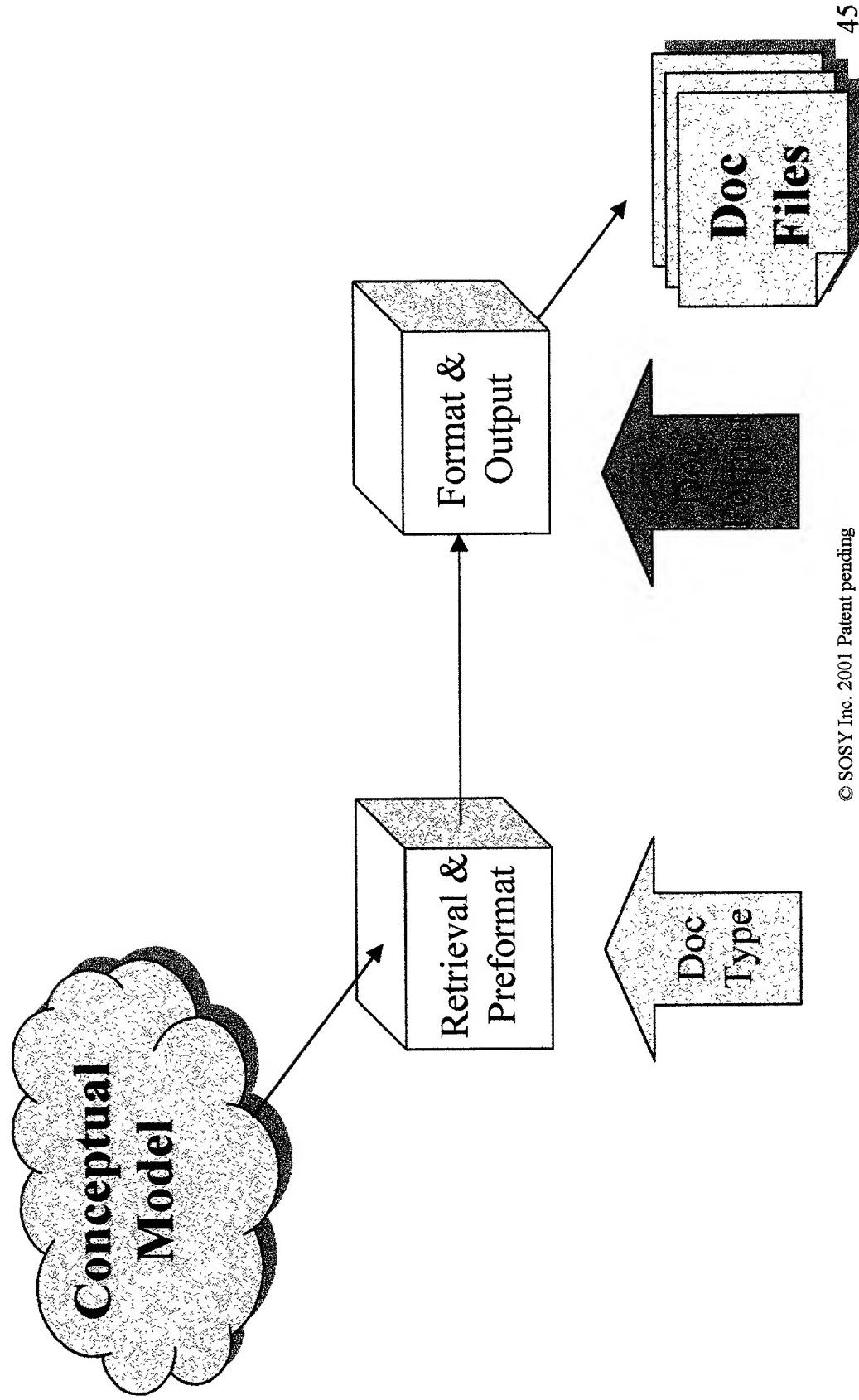
Translation

- Conceptual Model Subset of Interest
 - Subset of Interest depends on Document Type.
 - Usual elements:
 - Classes
 - Attributes
 - Relationships
 - Services & Arguments
 - Intensive use of analysis information.

Translation

- Translation Process
 - Read information from Conceptual Model and format it for output.
 - Two phases:
 - Information retrieval and pre-formatting.
 - Depends on Document Type
 - Independent from Document Format
 - Information output.
 - Depends on Document Format.
 - Independent from Document Type.

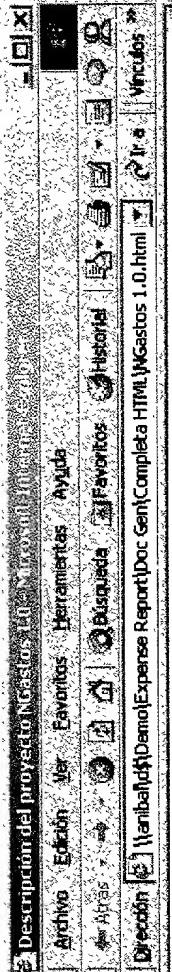
Translation Phases



Translation

- Remarks
 - Conceptual Model needs not to be valid (in terms of completeness and correctness) but it is always non-ambiguous.
 - The richer the analysis information, the richer the documentation.
 - Easily extensible
 - New Document Types
 - New Document Formats

Example



The screenshot shows a software interface with a title bar 'Descripción del proyecto Nro. 01'. The menu bar includes 'Archivo', 'Editar', 'Ver', 'Favoritos', 'Herramientas', and 'Ayuda'. Below the menu is a toolbar with icons for 'Nuevo', 'Abrir', 'Guardar', 'Copiar', 'Pegar', 'Borrar', 'Imprimir', and 'Ayuda'. The main area contains the following text:

Clase: Expense

Propiedades:

Alias: Expense

Observaciones: Employees may present a expense report when they have supported expenses on behalf of the company. (Agr Rel with Employee) Typically, the expenses are associated to a certain project or specific task. (Agr Rel with Project) At presenting the expense report, associated tickets will be attached and advances will be reflected. Advances must be discounted out from the expense report balance. The expense report, once presented, must be authorized by a responsible of the expenses. The authorization process will allow reject the expenses if proceed. Once authorized, the expense report will be approved for payment by a responsible of accounting. Once paid, it will be marked as so. (Agr Status)

Mensaje de Ayuda:

Hereda dc:

Se especializa en:

Se relaciona con: Employee, PaymentType, Project, MyCurrency, Lines

At the bottom right of the window, there is a status bar with 'Internet local'.

Persistence Relational Database Translation

CARE Technologies, S.A.

Index

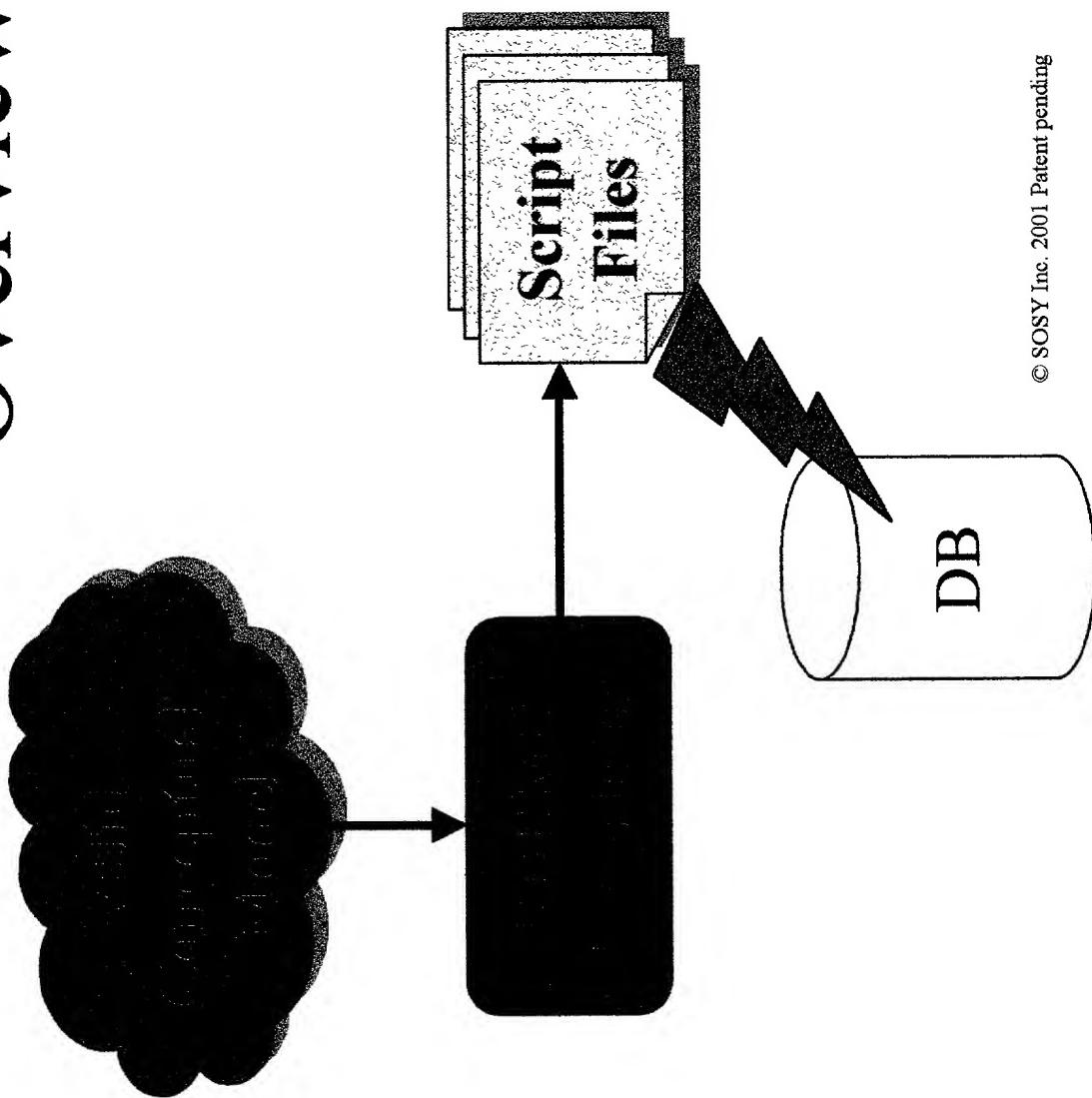
- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Persistence Relational Database Translation is the process of creating a Relational Database from a certain subset of information in the Object Model of a valid Conceptual Model.
- Output script files are used to create a relational database using structured query language (SQL).

Overview

- Creates
- Primary Keys
- Foreign Keys
- Indexes
- Drop Creates
- Drop Primary Keys
- Drop Foreign Keys
- Drop Indexes



Output Detail

- Creates
 - Creation of Tables and Fields
- Primary Keys
 - Creation of Primary Keys as constraints on each table
- Foreign Keys
 - Creation of Foreign Keys as constraints on each table
- Indexes
 - Creation of Indexed on each table

Output Detail

- Drop Creates
 - Deletion of Tables
- Drop Primary Keys
 - Deletion of Primary Key Constraints
- Drop Foreign Keys
 - Deletion of Foreign Key Constraints
- Drop Indexes
 - Deletion of Indexes

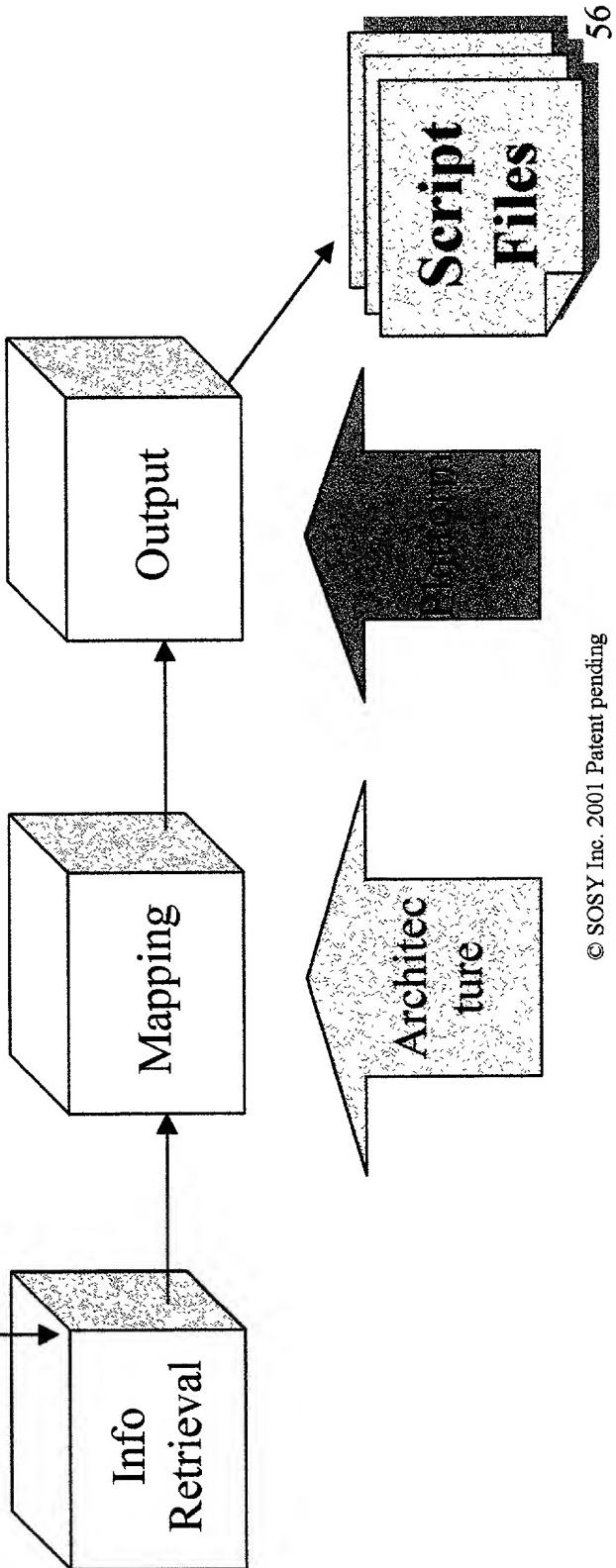
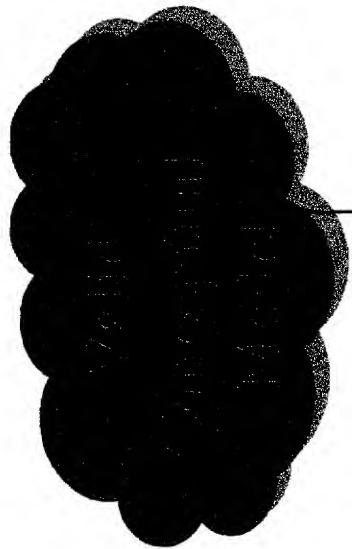
Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Classes
 - Attributes
 - Identification Functions
 - Aggregation Relationships
 - Inheritance Relationships

Translation

- Three phases:
 - Information retrieval.
 - Independent from persistence architecture.
 - Fixed architecture mapping.
 - Depends on persistence architecture.
 - Information output.
 - Targeted for Standard ANSI SQL 92 RDBMS.
 - Script files depends on the platform's SQL syntax of RDBMS manufacturer.
 - May depend on platform specifications to make use of manufacturer extensions and tuning.

Translation Phases



© SOSY Inc. 2001 Patent pending

Translation

- Translation Processes. Mapping:

- Class → Table
- Non-derived Attribute → Field
- Identification Function → Primary Key
- Univaluated Relationship → Foreign Key
- Univaluated Relationship → Index
- Multivaluated Relationship → Table
- Inheritance Relationship → Foreign Key

Example

Create table script in SQL for Expense class

```
CREATE TABLE Expense (
    fk_Project_1 int NOT NULL ,
    id_Expense int NOT NULL ,
    fk_Employee_1 CHAR(10) NOT NULL ,
    fk_MyCurrency_1 CHAR(5) NOT NULL ,
    fk_PaymentType_1 CHAR(5) NULL ,
    PresentDate datetime NOT NULL ,
    Status int NOT NULL ,
    Cause VARCHAR(255) NOT NULL ,
    AuthoDate datetime NULL ,
    AuthoComments VARCHAR(255) NULL ,
    PaymentDate datetime NULL ,
    PayComments VARCHAR(255) NULL ,
    Advances DECIMAL(19, 6) NOT NULL ,
    Exchange DECIMAL(19, 6) NOT NULL );
```

CARE Technologies, S.A.

Business Logic Translation

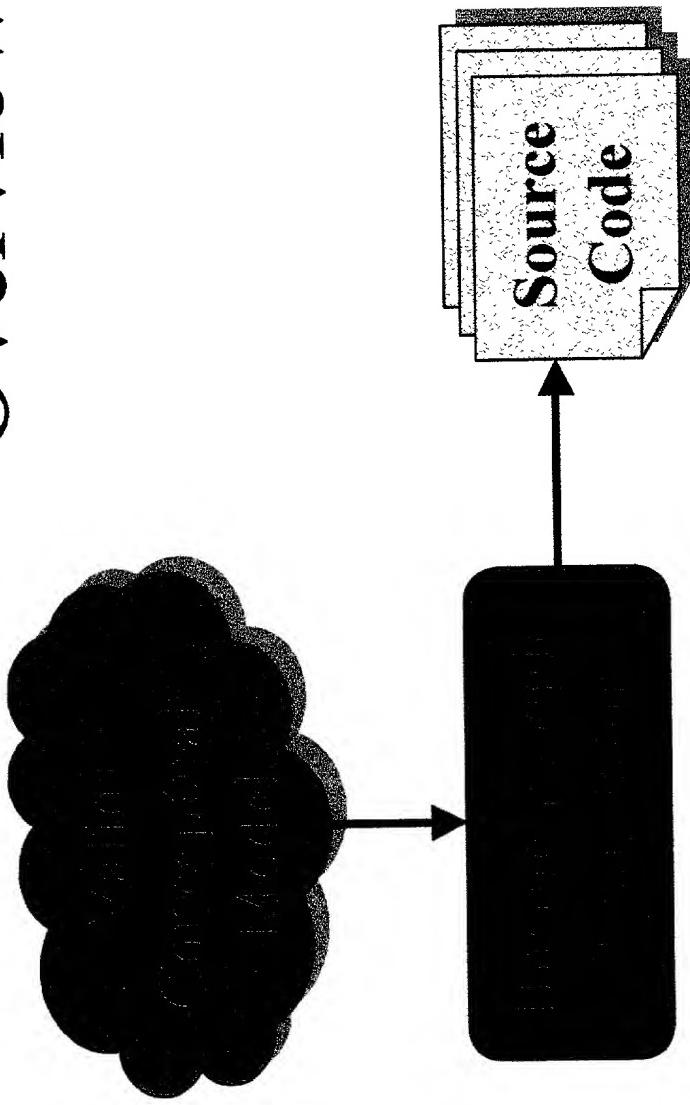
Index

- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Business Logic Translation is the process to obtain, following a precise Execution Model, the source code corresponding to the business logic from a valid Conceptual Model for a target Programming Language and Software Architecture.
- Execution Model is independent from Programming Language and Software Architecture.

Overview



Determines:

-*Target Programming Language*

-*Target Software Architecture*

© SOSY Inc. 2001 Patent pending

Output Detail

- Target Programming Language and Software Architecture determine:
 - Source code organization in files
 - Files internal organization
- Source Code's backbone: Execution Model.

Output Detail

- Traceability: Source code highly readable and maintainable thanks to:
 - Source code is always organized and structured in the same way.
 - Naming conventions applied.
 - Source code includes analysis information from the Conceptual Model as comments.

Output Detail

- Implementation of a precise Execution Model grants Functional Equivalence with Conceptual Model.
- Programming Interface to Clients for:
 - Actor Validation and Authentication.
 - Services Execution.
 - Queries Execution.
- Manages:
 - Concurrency.
 - Transactions.
 - Interoperable Objects Persistence.

Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Static properties (Visibility & Persistence)
 - Attributes + Identification Functions
 - Derivations
 - Aggregation Relationships
 - Inheritance Relationships
 - Services (Execution Model)
 - Arguments
 - Preconditions
 - Transaction Formulas
 - Actors (Execution Model)
 - Integrity Constraints (Execution Model)

© SOSY Inc. 2001 Patent pending

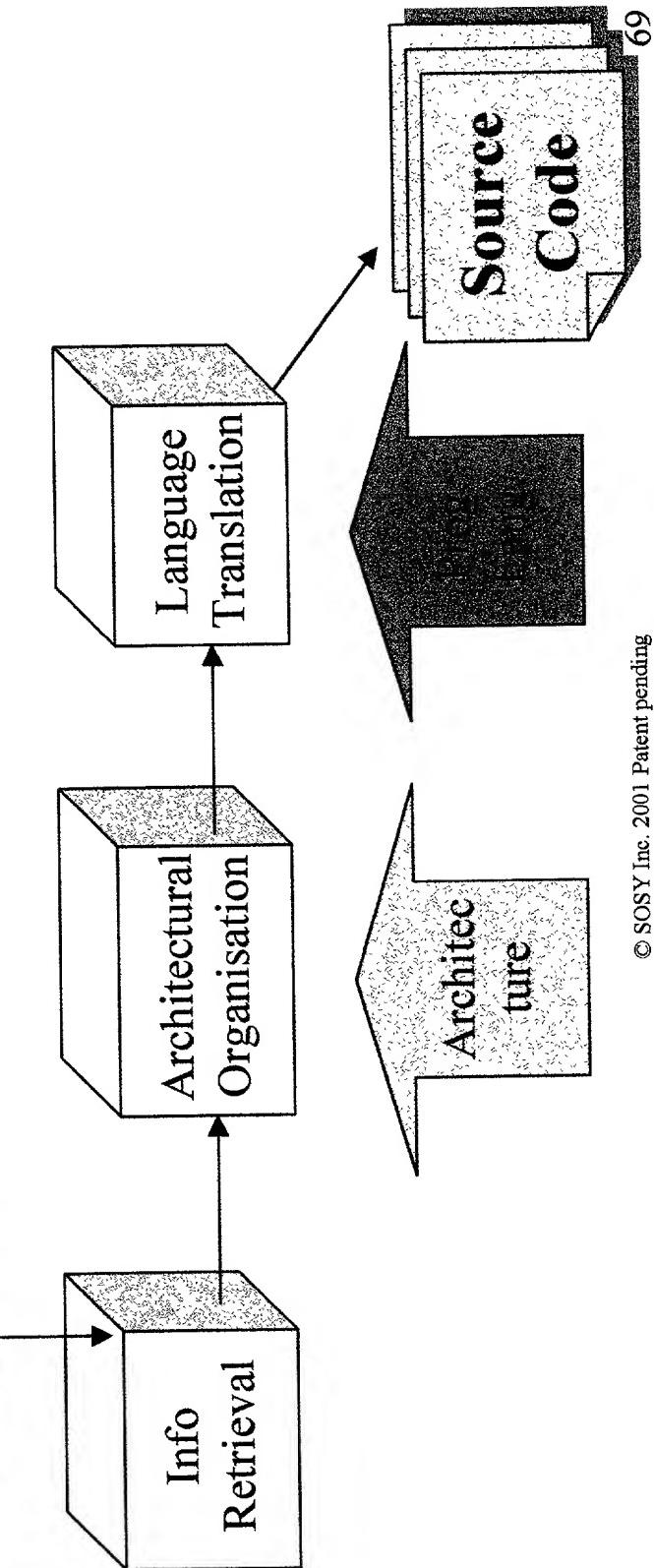
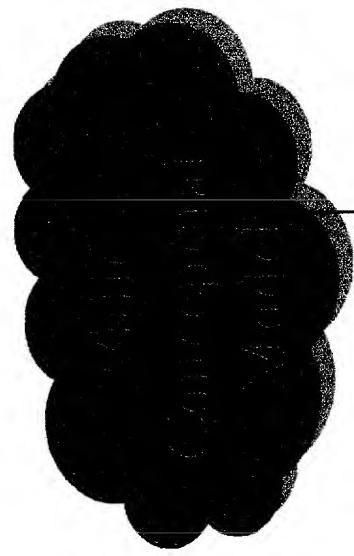
Translation

- Conceptual Model Subset of Interest.
 - Dynamic Model.
 - State Transition Diagram (Execution Model).
 - Controls Valid Lifes for an Object.
 - Object Interaction Diagram.
 - Triggers (Execution Model).
 - Global Transactions (Execution Model).
 - Functional Model (Execution Model).
 - Object state change upon occurrence of an event.

Translation

- Translation phases:
 - Information retrieval
 - Independent from target Software Architecture and Programming Language
 - Architectural organisation
 - Depends on target Software Architecture
 - Independent from target Programming Language
 - Determines files organisation and files internal structure
 - Language translation
 - Depends on target Programming Language
 - Influenced by Software Architecture
 - Takes advantage of Programming Language capabilities

Translation Phases



Translation

- Translation Processes

- Classes
 - Static properties translation
 - Services translation
 - Queries translation
- Global Interactions
 - Services translation
- Global Functions
 - Functions Interface translation
 - Body is left blank

Example

- Evaluation:
 - Service Authorize modifies attributes Status, AuthoDate and AuthoComments
 - Formal Specification Language expression for evaluation Valuation
 - [authorize ()] Status=2 and AuthoDate=today() and AuthoComments="" ,
 - Visual Basic Produced

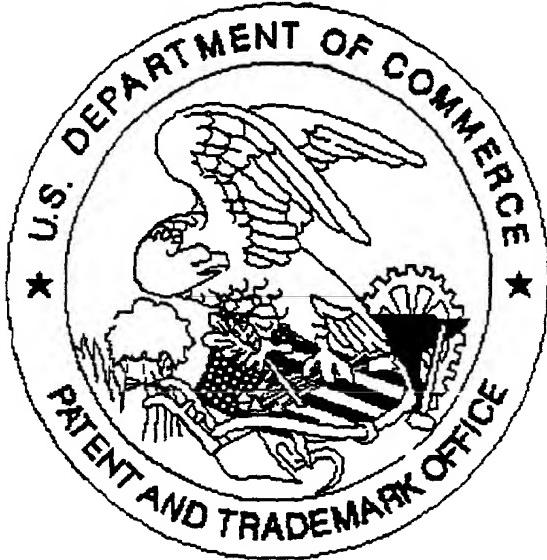
```
Private Function MV_Eval_Expense_authorize() As String  
Expense_Status = 2  
Expense_AuthoDate = today()  
Expense_AuthoComments = ""  
MV_Eval_Expense_authorize = ""  
End Function
```

© SOSY Inc. 2001 Patent pending

CARE Technologies, S.A.

User Interface Translation

United States Patent & Trademark Office
Office of Initial Patent Examination -- Scanning Division



Application deficiencies found during scanning:

Page(s) _____ of _____ were not present
for scanning. (Document title)

Page(s) _____ of _____ were not present
for scanning. (Document title)

Scanned copy is best available. Drawings and miscellaneous